Exact approaches to constrained submodular minimization problems

Modern Aspects of Submodularity workshop, Georgia Tech

March 19th-22nd, 2012

Kiyohito Nagano (University of Tokyo)

Joint work with Yoshinobu Kawahara
Satoru Iwata
Kazuyuki Aihara

Submodular minimization problems

f : submodular function defined on 2^N

poly time

- unconstrained minimization

$$\min_{S \subseteq N} f(S)$$

- minimization over
  - distributive lattice
  - parity family (Goemans-Ramakrishnan '95)

NP-hard

Zoya's talk

- size-constrained minimization

$$\min_{S \subseteq N} f(S) \quad \text{s.t.} \quad |S| = k$$

and other constrained problems

- partition (clustering) problem

$$\min \quad \sum_{i=1}^{k} f(S_i)$$

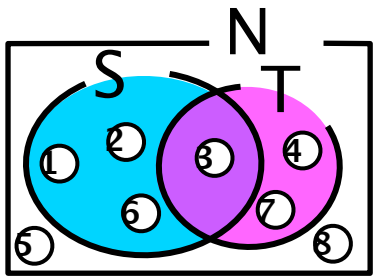
s. t. $\{S_1, \ldots, S_k\}$ is a partition of N (S_i is nonempty)

In this talk, we do not give approximation algorithms, but we give "exact" approaches to these NP-hard problems

Submodular function

- $N=\{1,\ldots,n\}$ is a finite set
- A function f defined on $2^N=\{S:S\subseteq N\}$ is **submodular** if $\forall S,T\subseteq N$

$$f(S)+f(T)\geq f(S\cup T)+f(S\cap T)$$



➢ arises in many fields, e.g.,
graph theory, game theory, economics, **machine learning**

Applications of submodular functions are extensively studied in recent years

Submodular function

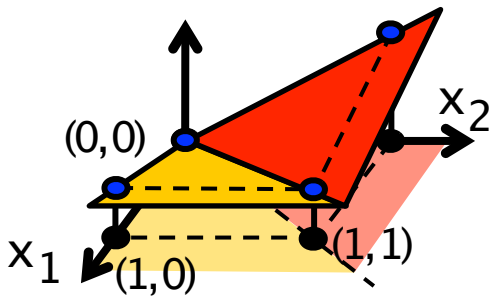
- **convex function** on $2^N \simeq \{0,1\}^n \quad (N=\{1,2,\dots,n\})$

A submodular function f ⇒ The interpolation function $\hat{f}$ is **convex**

The Lovász extension

$f(\{\})=0 \quad f(\{1\})=3$

$f(\{2\})=6 \quad f(\{1,2\})=3$



- **unconstrained submodular function minimization**

$$\min \ f(S) \quad \text{s.t.} \quad S \subseteq N$$

Given a value-giving oracle

- ellipsoid method: Grötschel-Lovász-Schrijver (1981, 1988)
- combinatorial algorithm: Schrijver (2000), Iwata-Fleischer-Fujishige (2000)

- "practical" algorithm: Fujishige-Wolfe algorithm (Fujishige '91, '05)

standard algorithm in ML

Examples of submodular functions

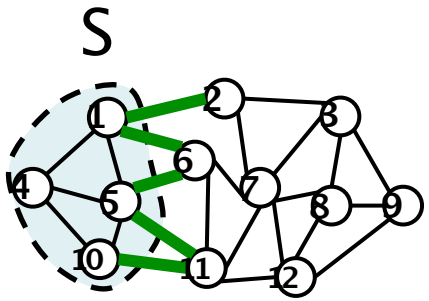
Graph $G = (N, E)$ with edge weights $w_e \geq 0$ $(e \in E)$

$N = \{1, \ldots, n\}$

- Ex. 1. **Cut function** $C : 2^N \to \mathbf{R}$

$$C(S) = \sum \{ w_e : \text{e∈E has one endpoint in S and one in N−S} \}$$

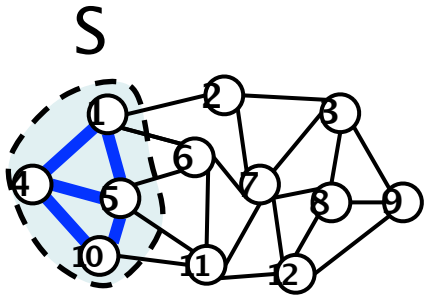
C is **submodular**



- Ex. 2. **Edge intensity function** $I : 2^N \to \mathbf{R}$

$$I(S) = \sum \{ w_e : \text{both of two endpoints of e∈E are in S} \}$$

I is **submodular** (i.e. I is **supermodular**)



Constrained submodular minimization

In many cases, constrained submodular minimization problems are hard (cf. Talks of Jan Vondrak on Monday)

In this talk, we give

- Exact approach to size-constrained submodular minimization (SSM)
  Nagano-Kawahara-Aihara, ICML '11

$$\min\ f(S) \quad \text{s.t.}\ |S| = k,\ S \subseteq N$$

➡ We use the minimum norm point of the base polyhedron

- Exact approach to partition problem with submodular objective function
  Nagano-Kawahara-Iwata, NIPS '10

$$\min\ \sum_{i=1}^{k} f(S_i) \quad \text{s.t.}\ \{S_1, \ldots, S_k\} \text{ is a partition of } N \ (S_i \text{ is nonempty})$$

We regard this problem as a clustering problem

➡ We use the theory of intersecting submodular functions

OUTLINE

- Introduction
- Exact approach to size-constrained submodular minimization (Nagano-Kawahara-Aihara, ICML'11)
  - Example: Densest subgraph problem
  - Size-constrained submodular minimization (**SSM**)
  - The algorithm through the minimum norm base
- Exact approach to clustering problems with submodular objective functions (Nagano-Kawahara-Iwata, NIPS'10)
  - Example: Clustering of a network (minimum k-cut)
  - Minimum Average Cost (MAC) clustering
  - The algorithm using the Dilworth truncation
- Remarks

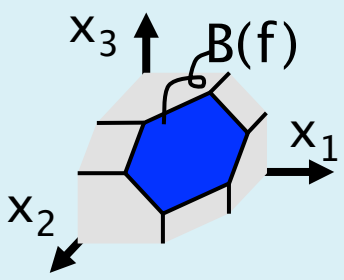
OUTLINE

Exact approaches to constrained submodular minimization problems

- Introduction
- Exact approach to size-constrained submodular minimization (Nagano-Kawahara-Aihara, ICML'11)
  - Example: Densest subgraph problem
  - Size-constrained submodular minimization (**SSM**)
  - The algorithm through the minimum norm base
- Exact approach to cluster

$f : 2^N \to \mathbf{R}$

$B(f) \subseteq \mathbf{R}^n$



$$\min \ f(S)$$
$$\text{sub. to } S \subseteq \{1,\dots,n\}, \ |S| = k$$

This problem generalizes
- densest k-subgraph problem
- size-constrained minimum cut problem

Fundamental NP-hard problems

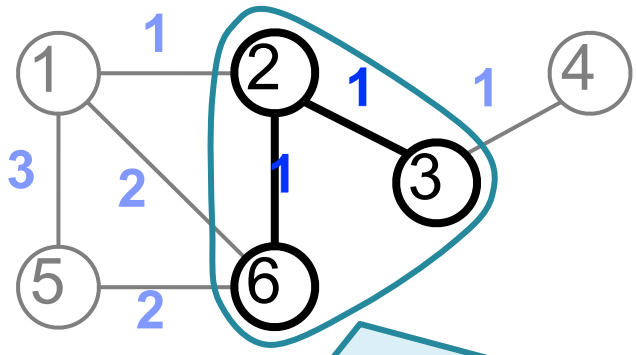
Example of **SSM**: Finding a dense graph

$G = (N, E)$: undirected graph

- node set $N = \{1, \ldots, n\}$
- edge set E

Edge weights $w_e \geq 0 (e \in E)$

Integer k $(0 \leq k \leq n)$



$S = \{2,3,6\}$

$\Rightarrow I(\{2,3,6\}) = 1 + 1 = 2$

Densest k-subgraph problem:

Find a k-subset $S \subseteq N$ ($|S| = k$) that maximizes I(S), where I(S) is sum of weights of edges within S

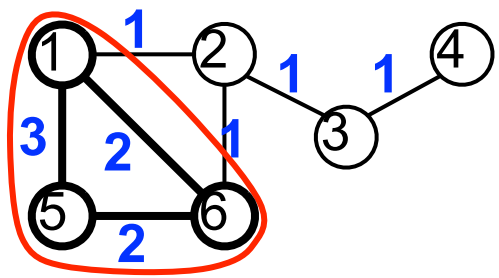
The set function $I: 2^N \rightarrow \mathbf{R}$ is supermodular (–I is submodular)

Example of **SSM**: Finding a dense graph

Densest k-subgraph problem:

maximize $I(S)$

subject to $S \subseteq \{1,\dots,n\}$, $|S| = k$



If $k = 3$, an optimal solution is $\{1,5,6\}$

- NP-hard and constant factor approx algorithms are not known

Applications:

- Community detection problem in complex networks
- Identification of functional modules in protein-protein interaction networks

We deal with a more general optimization problem

size-constrained submodular minimization

Unconstrained submodular minimization (USM)

Problem (USM)

$$\min \quad f(S)$$
$$\text{s.t.} \quad S \subseteq N = \{1, \ldots, n\}$$

solvable in poly time

- polynomial time:
  - ellipsoid method: Grotschel–Lovasz–Schrijver (1981, 1988)
  - combinatorial algorithm: Schrijver (2000), Iwata+ (2001)
- "**practical**": Fujishige–Wolfe algorithm (Fusjishige '91, '05)
  - **much faster in practice** (Fujishige+ 2006)
  - this algorithm computes the **minimum norm base**

    minimum L_2–norm point in the **base polyhedron** $B(f) \subseteq \mathbf{R}^n$
  - min norm base $\mathbf{x}^* \in \mathbf{R}^n$ ⇨ minimizer of $f \subseteq \{1, \ldots, n\}$

Size-constrained submodular minimization (SSM)

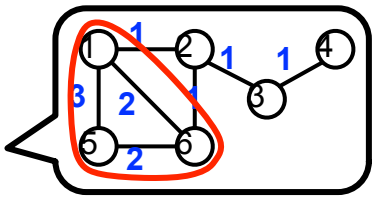
Problem (SSM)

$$\min \quad f(S)$$
$$\text{s.t.} \quad S \subseteq N,\ |S| = k$$

integer k $(0 \le k \le n)$

NP-hard

- There is no constant factor approximation algorithm that runs in polynomial time (Svitkina & Fleischer, '08&'11)
- **special cases**
  - $f = -|$ ⇨ Densest k-subgraph problem
  - $f = C$ ⇨ Size-constrained minimum cut problem



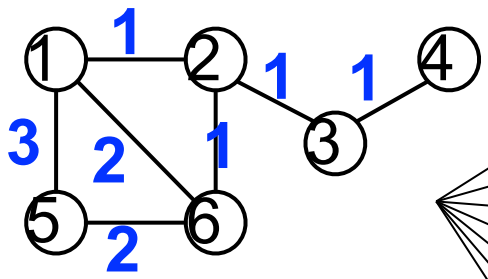
Both of these are fundamental NP-hard problems

We propose an new method for (SSM) that
- uses the **minimum norm base**
- computes "**a portion of exact optimal solutions**"

Example of outputs of the proposed method

For all k =0, … , n, consider the densest k-subgraph problems

max I(S) s.t. S⊆N, |S|=k



max I(S) s.t. S⊆N, |S|=0

max I(S) s.t. S⊆N, |S|=1

max I(S) s.t. S⊆N, |S|=2

max I(S) s.t. S⊆N, |S|=3

max I(S) s.t. S⊆N, |S|=4

max I(S) s.t. S⊆N, |S|=5

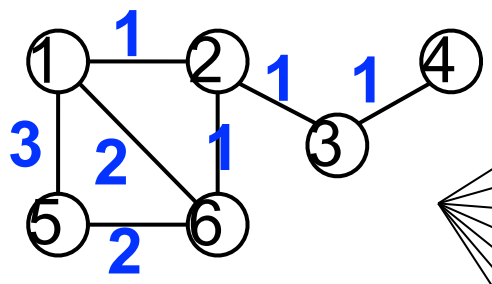
max I(S) s.t. S⊆N, |S|=6

n+1 optimization problems

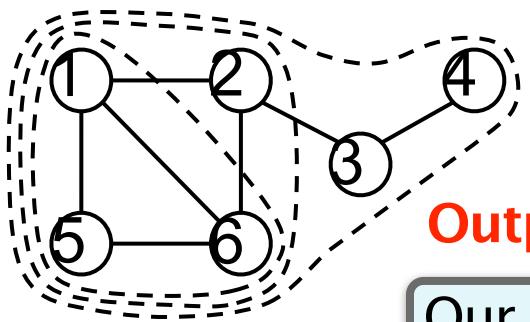
Example of outputs of the proposed method

For all k =0, … , n, consider the densest k-subgraph problems

max I(S) s.t. S⊆N, |S|=k



proposed method



max I(S) s.t. S⊆N, |S|=0 ➡ yes

max I(S) s.t. S⊆N, |S|=1 ➡ no

max I(S) s.t. S⊆N, |S|=2 ➡ no

max I(S) s.t. S⊆N, |S|=3 ➡ yes

max I(S) s.t. S⊆N, |S|=4 ➡ yes

max I(S) s.t. S⊆N, |S|=5 ➡ no

max I(S) s.t. S⊆N, |S|=6 ➡ yes

Outputs: {},{1,5,6},{1,2,5,6},{1,2,3,4,5,6}

Our method gives optimal solutions for k = 0, 3, 4, 6

Problem (SSM) in Machine Learning

$$\min f(S) \quad \text{s.t. } S \subseteq N,\ |S| = k$$

- **Densest k-subgraph problem** ($f = -I$)
  - This problem naturally formulates a **community detection** problem in complex networks
  - The identification of functional modules in **protein-protein interaction networks** is known as an important application (Dittrich et al., 2008)
- **Size-constrained minimum cut problem** ($f = C$)
  - This problem deals with an explicit size constraint

    Contrastingly, **spectral clustering**, which is one of the most popular clustering algorithms, can deal with a minimum cut problem with an implicit size constraint

Definition: Base polyhedron

For a submodular function $f : 2^N \to \mathbf{R}$ with $f(\{\}) = 0$,
the **base polyhedron** $B(f) \subseteq \mathbf{R}^n$ is given by

$$B(f) = \{\mathbf{x} \in \mathbf{R}^n : \sum_{i \in S} x_i \le f(S)\ (\forall S \subseteq N),\quad x_i \sum_{i \in N} = f(N)\}$$

$B(f)$ is determined by $2^n - 2$ inequalities and 1 equality

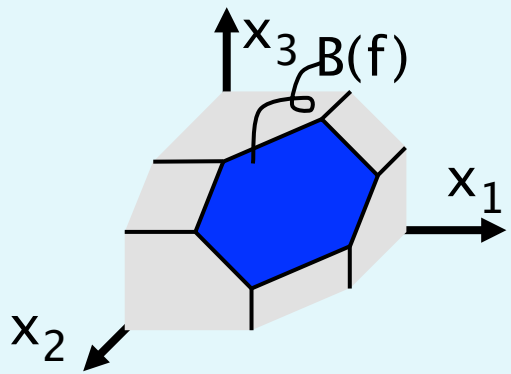
If $n = 3$, $B(f)$ is determined by

$$x_1 \le f(\{1\}),\quad x_1 + x_2 \le f(\{1,2\})$$
$$x_2 \le f(\{2\}),\quad x_1 + x_3 \le f(\{1,3\})$$
$$x_3 \le f(\{3\}),\quad x_2 + x_3 \le f(\{2,3\})$$

$2^3 - 2$ ineq

$$x_1 + x_2 + x_3 = f(\{1,2,3\})$$

1 equality



Minimum norm base and algorithms

The **minimum norm base** $\mathbf{x}^* \in \mathbf{R}^n$ is an optimal solution to

$$\min \sum_{i \in N} x_i^2 \quad \text{s.t.} \quad \mathbf{x} \in B(f)$$

- $\mathbf{x}^*$ can be computed efficiently (Fujishige'80; Fleischer–Iwata'03, Nagano'07)
- Fujishige–Wolfe algorithm ('91, '05) finds $\mathbf{x}^*$ **much faster in practice**

Algorithms through minimum norm base $\mathbf{x}^* \in \mathbf{R}^n$

- The Fujishige–Wolfe algorithm for (USM) $\min f(S)$ s.t. $S \subseteq N$

  $S^* = \{i \in N : x_i^* < 0\}$ minimizes f ⇨ The problem can be solved

  In FW algorithm, we use **partial information** about $\mathbf{x}^*$

- The **proposed algorithm** for (SSM) $\min f(S)$ s.t. $S \subseteq N$, $|S| = k$

  In our algorithm, we use **full information** about $\mathbf{x}^*$

Algorithm for (SSM)

$\min f(S)$ s.t. $S \subseteq N$, $|S| = k$

Consider the following algorithm:

surprisingly simple!

Algorithm SSM

Step1: Compute the minimum norm base $\mathbf{x}^* \in B(f) \subseteq \mathbf{R}^n$

Step2: Let $\xi_1 < \xi_2 < \cdots < \xi_d$ be distinct values of $\mathbf{x}^*$

Return $T_0 := \{\}$ and $T_j := \{i \in V : x_i \leq \xi_j^*\}, \forall j = 1, \ldots, d$

Example

The densest k-subgraph problems on $\Rightarrow B(-I) \subseteq \mathbf{R}^6$

The minimum norm base is $\mathbf{x}^* = (-\frac{7}{3}, -2, -1, -1, -\frac{7}{3}, \frac{7}{3}) \in B(-I)$

Thus we have $\xi_1 = -\frac{7}{3}$, $\xi_2 = -2$, $\xi_3 = -1$,

$T_0 = \{\}$, $T_1 = \{1,5,6\}$, $T_2 = \{1,2,5,6\}$, $T_3 = \{1,2,3,4,5,6\}$

optimal solutions for some of size constraints

Algorithm for (SSM)

$\min f(S)$ s.t. $S \subseteq V$, $|S| = k$

Consider the following algorithm:

surprisingly simple!

Algorithm SSM

Step1: Compute the minimum norm base $\mathbf{x}^* \in B(f) \subseteq \mathbf{R}^n$

Step2: Let $\xi_1 < \xi_2 < \cdots < \xi_d$ be distinct values of $\mathbf{x}^*$

Return $T_0 := \{\}$ and $T_j := \{i \in V : x_i \leq \overset{*}{\xi}_j\}, \forall j = 1, \ldots, d$

Theorem [This work] For each $j \in \{0, 1, \ldots, d\}$, $T_j \subseteq N$ is an optimal solution to Problem (SSM) w.r.t. $k = |T_j|$.

Algorithm SSM computes "a portion of exact optimal solutions"

Running time = computation of the minimum norm base

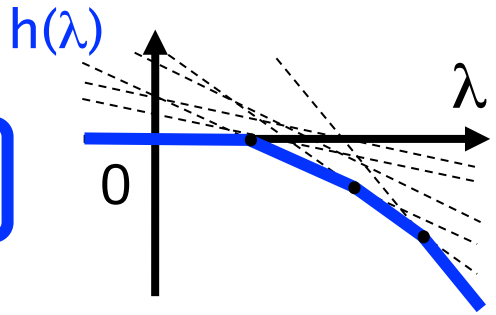
E.g., an implementation of the Fujishige–Wolfe algorithm can be found in a toolbox for submodular function optimization by Krause (2010, JMLR)

Proof of the validity of the algorithm SSM

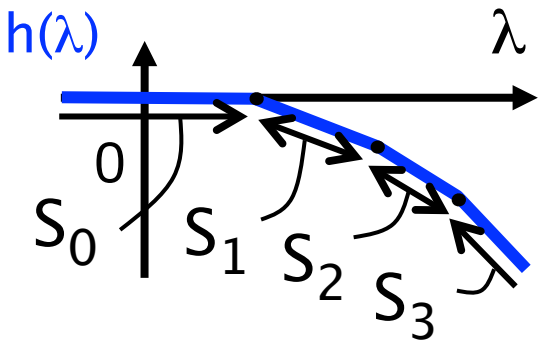
Define a function $h : \mathbf{R} \to \mathbf{R}$ as

$$h(\lambda) = \min\{f(S) - |S|\lambda : S \subseteq N\} \ (\lambda \in \mathbf{R})$$

linear function in λ



h is the minimum of 2^n linear functions



Each S_j is an **exact optimal solution** to problem (**SSM**) w.r.t. some size constraint

With the aid of the result of Fujishige (1980), we can show that $S_j = T_j$ for all j, where each T_j is the subset returned by the algorithm **SSM**

□

Application to synthetic data

- Networks are randomly generated from the GENRMF generator
- For each randomly generated network, we considered
  - the size-constrained minimum cut problem
  - the densest k-subgraph problem

The number of subsets (= d) found by the algorithm **SSM**:

dataset 1 (Genrmf-long)

n	cut	dense
63	2	50
126	2	101
256	2	213
525	2	466
1008	2	868

dataset 2 (Genrmf-wide)

n	cut	dense
75	2	68
147	2	136
324	2	298
576	2	530
1024	2	975

68 problems are exactly solved among 76 problems (68/76=89%)

 # problems exactly solved

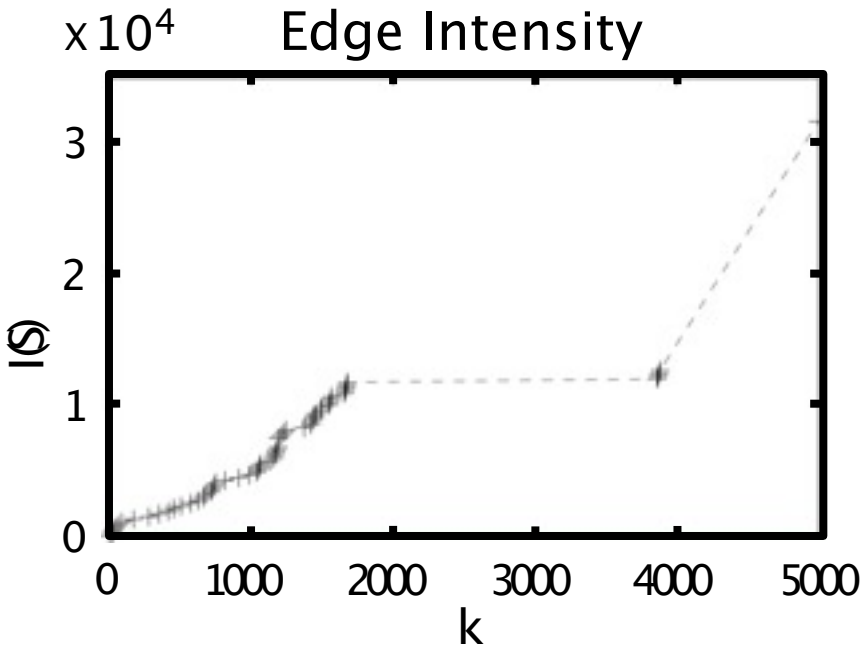
Application to real data

We applied the algorithm **SSM** to the densest k-subgraph problems on the network with **5,000** nodes and **31,664** edges (n = 5000)

This is a sub-network of social network data cnr-2000 (http://law.dsi.unimi.it/webdata/cnr-2000)

unweighted network

The algorithm provides optimal solutions for 57 out of 5000 size levels



OUTLINE

- Introduction
- Exact approach to size-constrained submodular minimization (Nagano-Kawahara-Aihara, ICML'11)
  - Example: Densest subgraph problem
  - Size-constrained submodular minimization (**SSM**)
  - The algorithm through the minimum norm base
- Exact approach to clustering problems with submodular objective functions (Nagano-Kawahara-Iwata, NIPS'10)
  - Example: Clustering of a network (minimum k-cut)
  - Minimum Average Cost (MAC) clustering
  - The algorithm using the Dilworth truncation
- Remarks

Clustering problems

clusters

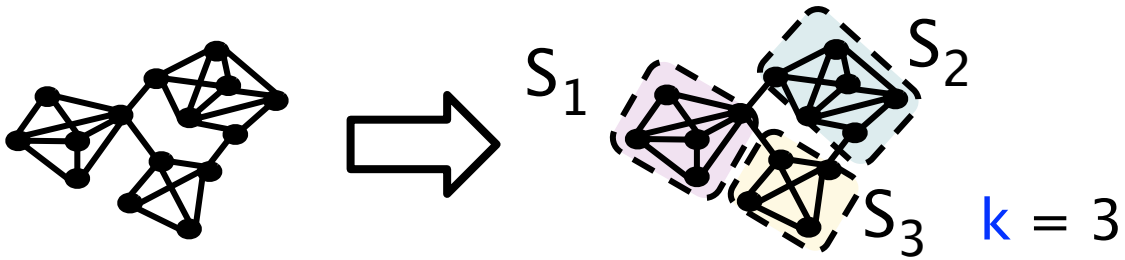
Given a set of data points N = {1, ... , n} and an integer k ≤ n,

find a partition of N into k subsets S_1, …, S_k ≠{} (called clusters) such that points in the same cluster are similar to each other and points in different clusters are dissimilar

- In many cases, k is not given in advance

Clustering methods

k-means, spectral clustering, maximum margin, etc



Example: clustering of networks

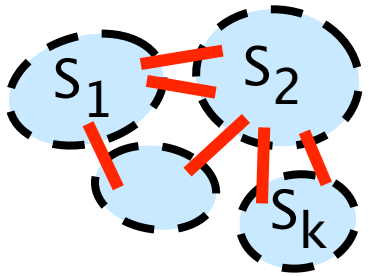
Example: clustering of a network

- Given a network G = (N, E) with nonegative edge weights
- Each edge indicates that its two endpoints are related

Find an "optimal" k-clustering

⇨ Compute a k-partition $\{S_1, \ldots, S_k\}$ that minimizes

sum of weights of red edges = $\frac{1}{2}\sum_{i=1}^{k} C(S_i)$



⇨ minimum k-cut (NP-hard)

Objective function:

$$\sum_{i=1}^{k} C(S_i) \xrightarrow{\text{generalize}} \sum_{i=1}^{k} f(S_i)$$

Narasimhan-Jojic-Bilmes, NIPS'05

f is a general submodular function

Clustering with submodular functions

$f : 2^N \to \mathbf{R}$ is a general submodular function with $f(\{\}) = 0$

■ **Optimal *k*-clustering problem** [Narasimhan *et al.*, NIPS '05]

$$\min \quad \sum_{i=1}^{k} f(S_i)$$
$$\text{s. t.} \quad \{S_1, \ldots, S_k\} \text{ is a } k\text{-partition of } N$$

- k should be computed via some method

☹ NP-hard

➢ generalizes clustering problems with some natural criteria

Precisely speaking, they dealt with a symmetric submodular function f and their clustering method is based on the algorithm of Queyranne (1998)

We will change the objective function

$$\sum_{i=1}^{k} f(S_i) \quad \Longrightarrow \quad \frac{\sum_{i=1}^{k} f(S_i)}{k-1}$$

k : fixed k : not fixed

Clustering with submodular functions

$f : 2^N \to \mathbf{R}$ is a general submodular function with $f(\{\}) = 0$

Optimal *k*-clustering problem [Narasimhan *et al.*, NIPS '05]

$$\min \quad \sum_{i=1}^{k} f(S_i)$$
$$\text{s. t.} \quad \{S_1, \ldots, S_k\} \text{ is a } k\text{-partition of } N$$

- k should be computed via some method

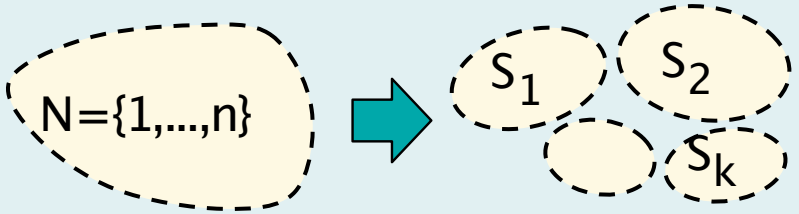
☹ NP-hard

Minimum Average Cost (MAC) clustering [This work]

$$\min \quad \sum_{S \in \mathcal{P}} f(S) \; / (|\mathcal{P}| - 1)$$
$$\text{s. t.} \quad \mathcal{P} \text{ is a partition of } N$$
$$|\mathcal{P}| > 1$$

Generalization using a parameter $\beta \in [0, n)$

This is a natural **average cost function**



$$\text{cost: } 0 \to \sum_i f(S_i)$$
$$\text{\# of sets: } 1 \to k$$
$$\Rightarrow \frac{\sum_i f(S_i)}{k-1}$$

Clustering with submodular functions

$f : 2^N \to \mathbf{R}$ is a general submodular function with $f(\{\}) = 0$

Optimal *k*-clustering problem [Narasimhan *et al.*, NIPS '05]

$$\min \quad \sum_{i=1}^{k} f(S_i)$$
$$\text{s. t.} \quad \{S_1, \ldots, S_k\} \text{ is a } k\text{-partition of } N$$

- k should be computed via some method

☹NP-hard

Minimum Average Cost (MAC) clustering [This work]

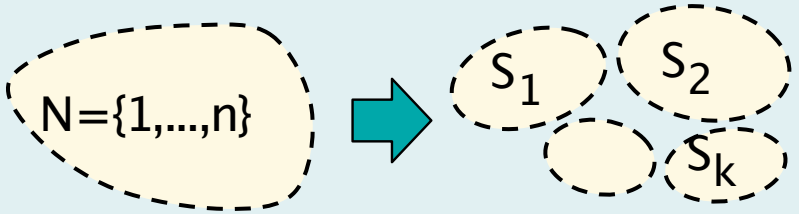
$$\min \quad \sum_{S \in \mathcal{P}} f(S) \,/ (|\mathcal{P}| - \beta)$$
$$\text{s. t.} \quad \mathcal{P} \text{ is a partition of } N$$
$$|\mathcal{P}| > \beta$$

where $0 \le \beta < n$

β – MAC clustering

If β is small, $\mathcal{P}$ is coarse
If β is large, $\mathcal{P}$ is fine

This is a natural **average cost function**



cost: $0 \to \sum_i f(S_i)$
# of sets: $1 \to k$

$$\left.\begin{array}{l}\text{cost: } 0 \to \sum_i f(S_i) \\ \text{\# of sets: } 1 \to k\end{array}\right\} \frac{\sum_i f(S_i)}{k-1}$$

Clustering with submodular functions

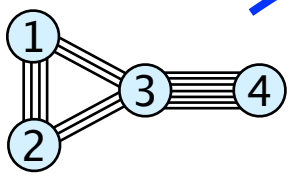
There is a relationship between the above two problems:

Prop. If a k-partition $\mathcal{P}$ is a β-MAC clustering for some β, then $\mathcal{P}$ is an optimal k-clustering

⇨ Information about MAC clusterings gives partial information about optimal k-clusterings

(Remember that an optimal *k*-clustering problem is NP-hard)

Example



Compute all β-MAC clusterings

$0 \le \beta < 1$	$1 \le \beta < 11/7$	$11/7 \le \beta < 4$
$\{\{1,2,3,4\}\}$	$\{\{1\},\{2\},\{3,4\}\}$	$\{\{1\},\{2\},\{3\},\{4\}\}$
opt 1-clustering	opt 3-clustering	opt 4-clustering

In this case, the MAC clustering algorithm computes optimal k-clusterings for k = 1, 3, 4

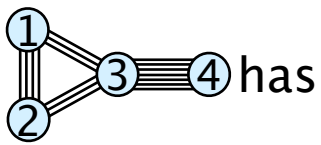
Let us see the framework of MAC clustering algorithm

Structure property of MAC clusterings (1/2)

Define a piecewise linear function $h : \mathbf{R}_+ \to \mathbf{R}$ as

$$h(\lambda) = \min\{ \sum_{S \in \mathcal{P}} f(S) - |\mathcal{P}|\lambda : \mathcal{P} \text{ is a partition of } N\} \ (\lambda \geq 0)$$

linear function in λ



has 15 partitions

1-p: {{1,2,3,4}} $0 - \lambda$

2-p: {{1,2},{3,4}}, {{1,3},{2,4}}, ... , {{4},{1,2,3}} $6 - 2\lambda$

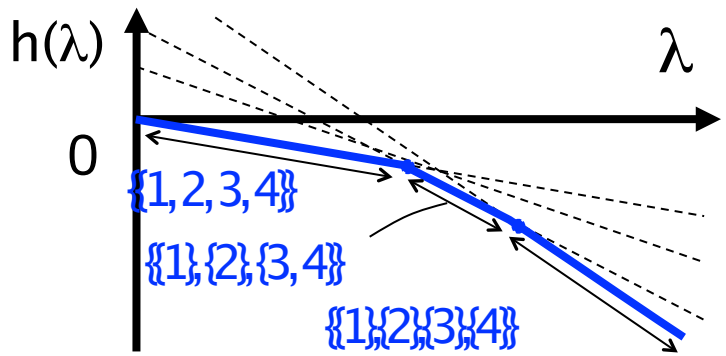
3-p: {{1},{2},{3,4}}, {{1},{3},{2,4}}, ... , {{3},{1,2}} $10 - 3\lambda$

4-p: {{1},{2},{3},{4}} $17 - 4\lambda$

Blue ones are opt k-clusterings

$h(\lambda) = \min\{-\lambda, 6-2\lambda, 10-3\lambda, 17-4\lambda\}$

{{1,2},{3,4}} does not determine h



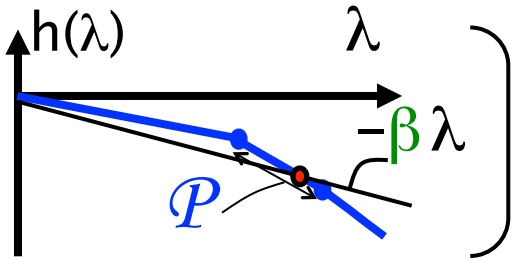
Structure property of MAC clusterings (2/2)

$$h(\lambda) = \min\{ \sum_{S \in \mathcal{P}} f(S) - |\mathcal{P}|\lambda : \mathcal{P} \text{ is a partition of } N\} \quad (\lambda \geq 0)$$

linear function in λ

If we know the structure of h, for any $\beta \in [0, n)$, a β –MAC clustering can be found immediately.

For $\beta \in [0, n)$, a partition $\mathcal{P}$ that corresponds to λ with $h(\lambda) = -\beta\lambda$ is a β –MAC clustering



Full information about the structure of the piecewise linear concave function h

⇩

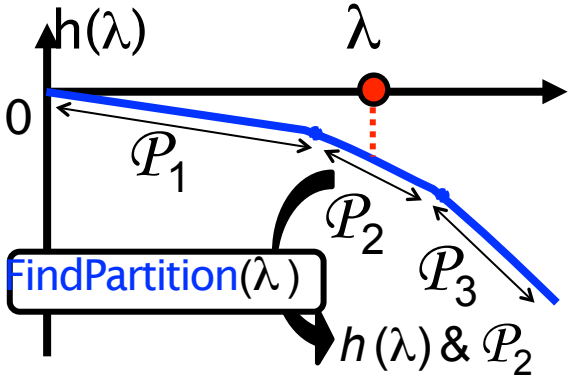
Full information about all the β – MAC clusterings

Framework of MAC clustering algorithm

(Full info about h) = (Partitions $\mathcal{P}_1,\ldots,\mathcal{P}_d$ that determines h)

- The MAC algorithm uses

Procedure FindPartition(λ) :
For λ ≥ 0, the procedure computes h(λ) & a partition $\mathcal{P}$ that determines h at λ



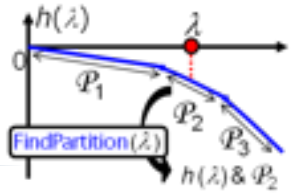
With the aid of the Dilworth truncation, this procedure runs in O(n · SFM (*n*)) time

unconstrained submodular minimization

- By performing FindPartition O(n) times, we obtain all $\mathcal{P}_j$

Theorem [This work] Full information about all β–MAC clusterings can be computed in $O(n^2 \cdot SFM(n))$ time

Procedure FindPartition(λ)



This procedure solves

$$h(\lambda) = \min_{S \in \mathcal{P}} \sum (f(S) - \lambda)$$
$$\text{s.t. } \mathcal{P} \text{ is a partition of N}$$

Define $f^\lambda: 2^N \to \mathbf{R}$ as $f^\lambda(S) = \begin{cases} f(S) - \lambda & \text{if S is nonempty} \\ 0 & \text{if S is empty} \end{cases}$

➡ f^λ is an intersecting submodular function

Let $f_p^\lambda: 2^N \to \mathbf{R}$ be the **Dilworth truncation** (1944) of f^λ

$$\left(f_p^\lambda(S) = \min\{ \sum_{S \in \mathcal{P}} f^\lambda(S) : \mathcal{P} \text{ is a partition of } S \} \right)$$

➡ Then, $h(\lambda) = f_p^\lambda(N) = \min\{ \sum_{S \in \mathcal{P}} f^\lambda(S) : \mathcal{P} \text{ is a partition of N} \}$

An optimal paritition can be found in $O(n \cdot \text{SFM}(n))$ time

See Chapter48 of Schrijver's book [2003]

Experimental results

We compared the **MAC** algorithm with
(1) k-means method
(2) spectral-clustering method with normalized-cut (Ng+, 2002)
(3) maximum-margin clustering (Xu+, 2005)

# of clusters k is selected through 5-fold cross-validation

360 samples

	Gaussian	Circle	Iris	Libras	Glass
k-means	**1.0**	0.88	0.79	0.85	0.93
norm cut	0.88	0.86	0.84	0.87	0.93
max margin	0.99	**1.0**	0.96	0.90	**0.97**
MAC	0.99	**1.0**	**0.99**	**0.97**	**0.97**

'Gaussian', 'Circle': synthetic datasets

'Iris', 'Libras', 'Glass': real world datasets from the UCI repository

Clustering accuracy (the larger, the better)

MAC clustering is competitive with standard clustering methods

Remarks

- Submodular optimization problems are extensively studied in machine learning
- The proposed algorithms run in polynomial time
- The results contrast sharply with the NP–hardness of the problems
- Minimization of the difference of two submodular functions is also an important problem in ML

$$\min_{S \subseteq N} f_1(S) - f_2(S)$$

Narasimhan–Bilmes (UAI'05),
Kawahara–Washio (NIPS '11)

↖ Discrete version of DC programming

END

Appendix: Description of FindPartition

$V = \{1, \dots, n\}$, $V^m = \{1, \dots, m\}$ for each $m = 1, \dots, n$

$\boldsymbol{e}_m$ is the m-th unit vector $\in \mathbf{R}^n$ for each $m = 1, \dots, n$

Procedure FindPartition (λ)

Input: A real value $\lambda \geq 0$

Output: A real value $h(\lambda)$ and a partition $\mathcal{P}_\lambda$ of V

S1: Set $\mathcal{P}^0 := \{\}$.

Set $\boldsymbol{x}^0 := -M \bullet \mathbf{1}_n = (-M, \dots, -M)$ for some large $M > 0$.

S2: For each $m = 1, \dots, n$, do

Compute $\alpha^m = \min \{ f(S) - \boldsymbol{x}^{m-1}(S) - \lambda : m \in \forall S \subseteq V^m \}$
and $T^m = \operatorname{argmin} \{ f(S) - \boldsymbol{x}^{m-1}(S) - \lambda : m \in \forall S \subseteq V^m \}$.

SFM

Set $\boldsymbol{x}^m := \boldsymbol{x}^{m-1} + \alpha^m \bullet \boldsymbol{e}_m$,
set $U^m := T^m \cup [\cup \{S : S \in \mathcal{P}^{m-1},\ T^m \cap S \neq \{\}\}]$, and
set $\mathcal{P}^m := \{U^m\} \cup \{S : S \in \mathcal{P}^{m-1},\ T^m \cap S = \{\}\}]$.

S3: Return $h_\lambda := \boldsymbol{x}^n(V)$ and $\mathcal{P}_\lambda := \mathcal{P}^n$.
